CMPT 250
Midterm 2

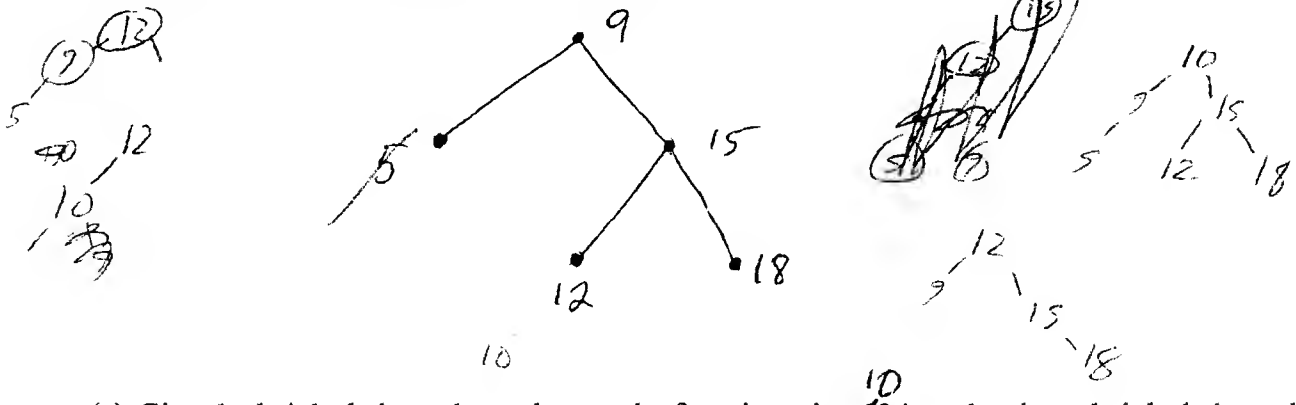
55 minutes | Closed Book | Mar. 17, 2006
Marks

4 1. For the values (characters) and activity counts given in the following table, give the weight-balanced binary tree that stores the values.

Value	Activity count
'a'	2
'f'	5
'm'	3
'p'	1

4 2. For most data structures, the search time is only related to the number of values presently in the data structure. However, for an open address hash table, values previously deleted can slow down the search time. Explain why this is true.

6 3. Consider the following height-balanced tree:

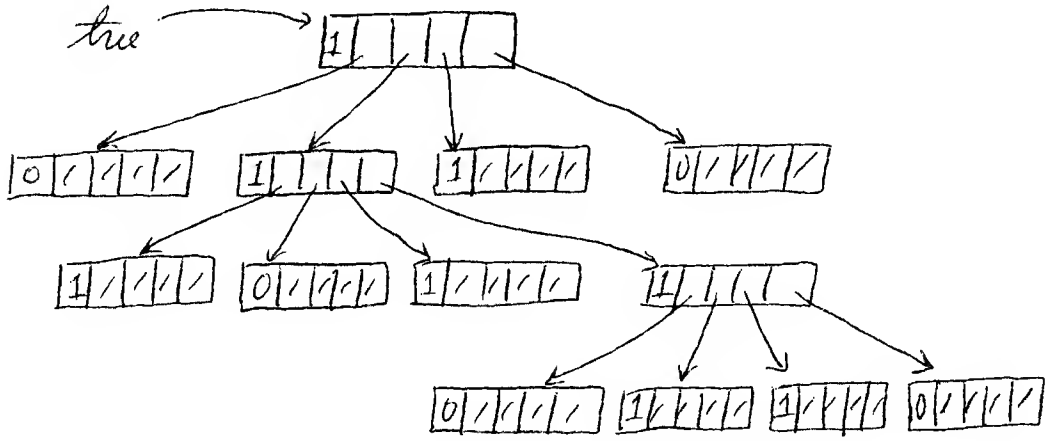


(a) Give the height-balanced tree that results from inserting 12 into the above height-balanced tree.

(b) Give the height-balanced tree that results from deleting 5 from the above height-balanced tree. Note that the deletion is from the height-balanced tree shown above, not from the answer to part (a).

6 4. Consider the comparison of a height-balanced tree and an open-address hash table. Select one of them, and explain when it should be used as compared with the other one.

8 5. A tree is said to be a quad-tree if every node has either 0 or 4 children. An example of a quad-tree storing zeroes and ones is given below. **Prove by mathematical induction** that a non-empty quad-tree with i interior nodes has a total of 4*i + 1 nodes. An interior node is a node with 4 children.



7 6. Give the situations to be tested for the following routine from the class LINKED_SIMPLE_TREE_UOS [G]:

```
out_with_level_numbers (i : INTEGER) : STRING is
    -- Form an inorder string representation that includes level number
    -- Analysis: Time = O(n), n = number of items in the (sub)tree
  local
    blanks: STRING
  do
    Result := ""
    if not is_empty and then
        not (root_left_subtree.is_empty and root_right_subtree.is_empty) then
      Result := Result + root_right_subtree.out_with_level_numbers(i+1)
    end
    create blanks.make((i-1)*5)
    blanks.fill_blank
    Result := Result + "%N" + blanks + i.out + ": "
    if is_empty then
      Result := Result + "-"
    else
      Result := Result + root_item.out
      if not (root_left_subtree.is_empty and root_right_subtree.is_empty) then
        Result := Result + root_left_subtree.out_with_level_numbers(i+1)
      end
    end
  end
```

If useful, the features of the class LINKED_SIMPLE_TREE_UOS [G] are given in the next question.

6 7. Consider the following grammar for an expression

<A> ::= 7 | 5 # <A>

where # is the operator.

(a) Give an example of a string with length at least 5 that matches this grammar.

(b) Give the pseudo-code (general algorithm) for the outline of a routine to use recursive-descent parsing to parse sentences that fit this grammar. The routine does not do anything except read the appropriate characters and do appropriate calls in the correct sequence.

9 8. Give the Eiffel code for a function in a class for an ordered binary tree where the function has one parameter, call it x. The function is to find the sum of all items in the tree whose values are at least as large as x. Assume that the class is a descendant of the class LINKED_SIMPLE_TREE_UOS [G], but the tree is kept ordered. Where possible, the function should take advantage of the ordering of the tree. Note that you only need to give the code for the function, not the whole descendant class.

For your reference, the features of the class LINKED_SIMPLE_TREE_UOS [G] are the following:

```
make                                initialize (lt : like Current; x : G; rt : like Current)
root_item : G                           out : STRING
is_empty : BOOLEAN                      is_full : BOOLEAN
root_left_subtree : like Current        root_right_subtree : like Current
```

Total 50 The end